

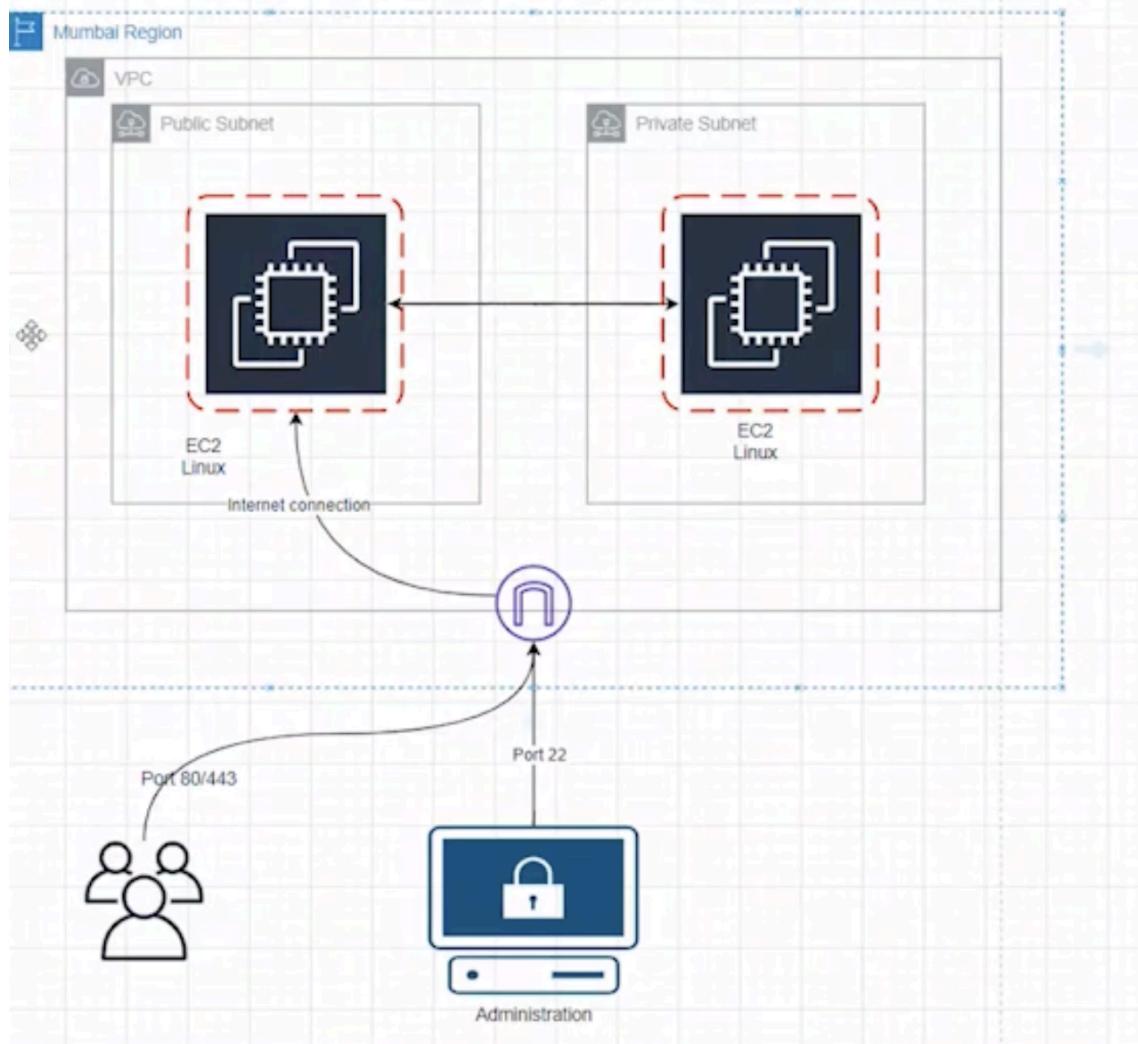
AWS VPC Hands-On: Public and Private EC2 with Internet Gateway

→ create networking on AWS which includes AWS VPC, Subnets, Internet Gateway, Route Tables, Security Groups along with a server inside the network.

➤ TASK: created a custom AWS VPC with one public EC2 and one private EC2. The public EC2 has internet access using an Internet Gateway, and I used it as a jump server to connect to the private EC2, which has no direct internet access

ADVANTAGES:

To allow internet access only where required and keep backend servers safe.



Step 1: Select Mumbai Region

Step 2: Create vpc

Name: Demo-vpc

The screenshot shows the AWS VPC console with a success message: "You successfully created vpc-Ofb771a02f54d53ca / Demo-vpc". The VPC details page is displayed, showing the following configuration:

VPC ID	State	Block Public Access	DNS hostnames
vpc-Ofb771a02f54d53ca	Available	Off	Disabled
DNS resolution	Tenancy	DHCP option set	Main route table
Enabled	default	dopt-0af5547b670e1321b	rtb-02029a6356fca4d4f
Main network ACL	Default VPC	IPv4 CIDR	IPv6 pool
acl-0d4b9813222974de7	No	10.0.0.0/16	-
IPv6 CIDR (Network border group)	Network Address Usage metrics	Route 53 Resolver DNS Firewall rule groups	Owner ID
-	Disabled	-	801458814717
Encryption control ID	Encryption control mode	-	
-	-		

Below the main details, there are tabs for "Resource map", "CIDRs", "Flow logs", "Tags", and "Integrations".

Step3 : Create Subnets

Go to VPC → Subnets → Create subnet

Select your VPC

Subnet name: public-subnet-az-1

Availability Zone: ap-south-1a

CIDR block: 10.0.0.0/24

Add a subnet

subnet name: private-subnet-az-2

Availability Zone: ap-south-1b

CIDR block: 10.0.1.0/24

Click Create subnet

The screenshot shows the AWS VPC dashboard with the "Subnets" section selected. It displays three subnets:

Name	Subnet ID	State	VPC	Block Public...	IPv4 C
-	subnet-0f267fd671d9f0c20	Available	vpc-03e487a9440b8ff6b MYVPC	Off	172.31
public-subnet-az-2	subnet-007d7b77ec9f2370e	Available	vpc-Ofb771a02f54d53ca Dem...	Off	10.0.1
public-subnet-az-1	subnet-0f0c85ac138a85f07	Available	vpc-Ofb771a02f54d53ca Dem...	Off	10.0.0

Below the table, the details for "public-subnet-az-1" are shown:

IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association ID
10.0.0.0/24	-	-
Availability Zone	VPC	Route table
ap-s1-az1 (ap-south-1a)	vpc-Ofb771a02f54d53ca Demo-vpc	rtb-02029a6356fca4d4f
Network ACL	Auto-assign public IPv4 address	Auto-assign IPv6 address
acl-0d4b9813222974de7	No	No
Auto-assign customer-owned IPv4 address	Outpost ID	IPv4 CIDR reservations
No	-	-
Customer-owned IPv4 pool		
-		

Name	Subnet ID	State	VPC	Block Public...	IPv4
-	subnet-0f267fd671d9f0c20	Available	vpc-03e487a9440b8ff6b MYVPC	Off	172.3
<input checked="" type="checkbox"/> public-subnet-az-2	subnet-007d7b77ec9f2370e	Available	vpc-0fb771a02f54d53ca Demo...	Off	10.0.1
<input type="checkbox"/> public-subnet-az-1	subnet-0fc85ac138a85f07	Available	vpc-0fb771a02f54d53ca Demo...	Off	10.0.0

subnet-007d7b77ec9f2370e / public-subnet-az-2	
<input type="checkbox"/> subnet-007d7b77ec9f2370e	<input type="checkbox"/> arn:aws:ec2:ap-south-1:801458814717:subnet/subnet-007d7b77ec9f2370e
IPv4 CIDR	<input type="checkbox"/> Available
<input type="checkbox"/> 10.0.1.0/24	<input type="checkbox"/> IPv6 CIDR
Availability Zone	<input type="checkbox"/> -
<input type="checkbox"/> aps1-az3 (ap-south-1b)	IPv6 CIDR association ID
Network ACL	<input type="checkbox"/> -
acl-0d4b9813222974de7	Route table
Auto-assign customer-owned IPv4 address	<input type="checkbox"/> rtb-02029a6356fca4d4f
No	Auto-assign IPv6 address
No	No
No	IPv4 CIDR reservations
No	-

Step 4: Create Internet Gateway

Go to VPC → Internet Gateways

Click Create internet gateway

Name: Demo-vpc-igw

Click Create internet gateway

The screenshot shows the AWS VPC Internet Gateways page. A new Internet Gateway is being created with the following details:

- Internet gateway ID:** igw-01275d537a5335b71
- State:** Attached
- VPC ID:** vpc-0fb771a02f54d53ca | Demo-vpc
- Owner:** 801458814717
- Tags (1):** Name: Demo-vpc-igw

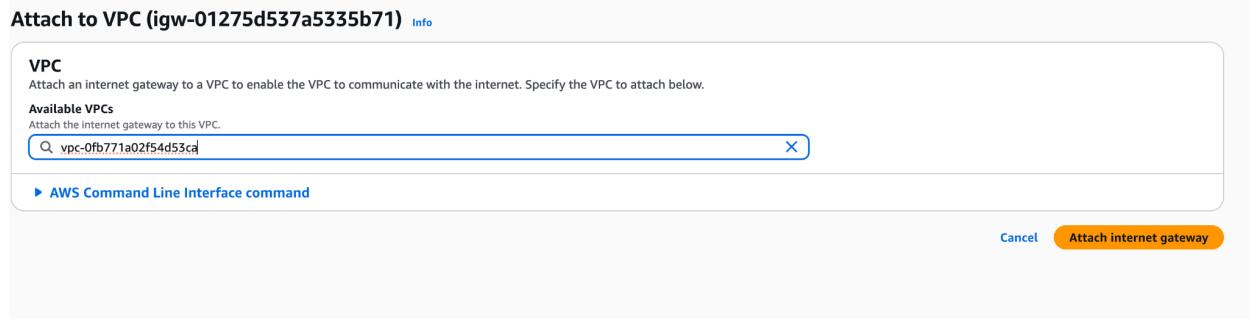
Attach Internet Gateway to VPC

Select the Internet Gateway

Click Actions → Attach to VPC

Select your VPC (Demo-vpc)

Click Attach



→ Both subnets having same route table ID. we need to change the route table .Because if we change any rule in any route table it will effect both subnets

Step 4: create a Route table

Go to VPC → Route Tables

Click Create route table

Name: public-rt-demo-vpc

Select: Demo-vpc

Create it

While we creating the subnet's by default one route table will create (ex route id : [rtb-02029a6356fca4d4f](#)) this is route id for both private and public subnet. Because of the we created new route table. Now in Route table we have two route table one is default another one is which we created. Default one name it has private-rt-demo-vpc

Route tables (1/4) [Info](#)

Last updated 4 minutes ago [Actions](#) [Create route table](#)

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
<input checked="" type="checkbox"/> private-rt-demo-vpc	rtb-02029a6356fca4d4f	-	-	Yes	vpc-0fb771a02f54d53
<input type="checkbox"/> -	rtb-0470115e644fe56c7	-	-	Yes	vpc-007645041d5203
<input type="checkbox"/> -	rtb-0def12281117fa121	-	-	Yes	vpc-03e487a9440b8ff
<input type="checkbox"/> public-rt-demo-vpc	rtb-0361fdb8b74ab3f44	-	-	No	vpc-0fb771a02f54d53

[rtb-02029a6356fca4d4f / private-rt-demo-vpc](#)

[Details](#) [Routes](#) [Subnet associations](#) [Edge associations](#) [Route propagation](#) [Tags](#)

Details

Route table ID rtb-02029a6356fca4d4f	Main <input checked="" type="checkbox"/> Yes	Explicit subnet associations -	Edge associations -
VPC vpc-0fb771a02f54d53ca Demo-vpc	Owner ID 801458814717		

<input type="checkbox"/> -	rtb-0def12281117fa121	-	-	Yes	vpc-03e487a9440b8ff
<input checked="" type="checkbox"/> public-rt-demo-vpc	rtb-0361fdb8b74ab3f44	-	-	No	vpc-0fb771a02f54d53

[rtb-0361fdb8b74ab3f44 / public-rt-demo-vpc](#)

[Details](#) [Routes](#) [Subnet associations](#) [Edge associations](#) [Route propagation](#) [Tags](#)

Details

Route table ID rtb-0361fdb8b74ab3f44	Main <input type="checkbox"/> No	Explicit subnet associations -	Edge associations -
VPC vpc-0fb771a02f54d53ca Demo-vpc	Owner ID 801458814717		

(*** now both are have different route id's)

→ Every route table having default IP that is 10.0.0.0/16. If vpc wants to communicate internally to router at that it will use this 10.0.0.0/16. By default every router will have this ip this is for internal communication

[rtb-02029a6356fca4d4f / private-rt-demo-vpc](#)

[Details](#) [Routes](#) [Subnet associations](#) [Edge associations](#) [Route propagation](#) [Tags](#)

Routes (1)

[Filter routes](#)

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	Create Route Table

Step 5: Now private route table having Two subnet's now we need to detach one subnet

Go to private-rt-demo-vpc ,--> go to subnet association (there we have public and private subnets attached)

→for private subnet select private route and save associations

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/2)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
<input checked="" type="checkbox"/> private-subnet-az-2	subnet-007d7b77ec9f2370e	10.0.1.0/24	-	Main (rtb-02029a6356fca4d4f / privat...)
<input type="checkbox"/> public-subnet-az-1	subnet-0fc85ac138a85f07	10.0.0.0/24	-	Main (rtb-02029a6356fca4d4f / privat...)

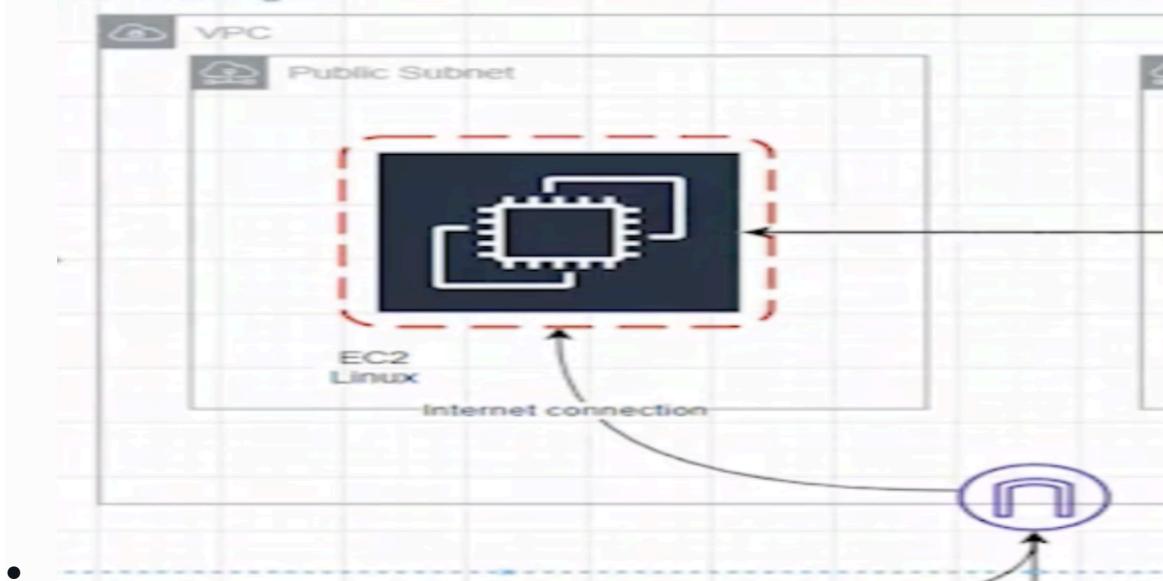
Selected subnets

subnet-007d7b77ec9f2370e / private-subnet-az-2 X

Cancel Save associations

→for public subnet select public route

- Need internet connection for public route for that need to add the rule for public route table . for the internet connect ip is 0.0.0.0/0



VPC > Route tables > rtb-0361fdb8b74ab3f44 > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
Q_ 0.0.0.0/0	X	-	No	CreateRoute
Q_ igw-01275d537a5335b71	X	-		

[Add route](#)

[Cancel](#) [Preview](#) [Save changes](#)

Subnet	Route Table	Status	Propagation	IP Range
private-subnet-az-2	subnet-007d7b77ec9f2370e	Available	vpc-0fb771a02f54d53ca Dem...	10.0.1
<input checked="" type="checkbox"/> public-subnet-az-1	subnet-0f0c85ac138a85f07	Available	vpc-0fb771a02f54d53ca Dem...	10.0.0

subnet-0f0c85ac138a85f07 / public-subnet-az-1

Route table: rtb-0361fdb8b74ab3f44 / public-rt-demo-vpc

[Edit route table association](#)

Routes (2)	
Q Filter routes	
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-01275d537a5335b71

Save the changes

Need to edit the public route for the public subnet we need to have public route only.

→ Go to public-rt-demo-vpc , → go to subnet association (there we have public and private subnets attached) select the public-subnet-az-1 and save the changes

<input type="checkbox"/> private-rt-demo-vpc	rtb-02029a6356ca4dd4	subnet-007d7b77ec9f23...	-	Yes	vpc-0fb771a02f54d
<input checked="" type="checkbox"/> public-rt-demo-vpc	rtb-0361fdb8b74ab3f44	subnet-0f0c85ac138a85f...	-	No	vpc-0fb771a02f54d
<input type="checkbox"/> -	rtb-0470115e644fe56c7	-	-	Yes	vpc-007645041d521
<input type="checkbox"/> -	rtb-0def12281117fa121	-	-	Yes	vpc-03e487a9440bf

rtb-0361fdb8b74ab3f44 / public-rt-demo-vpc

Details | Routes | **Subnet associations** | Edge associations | Route propagation | Tags

Explicit subnet associations (1)

[Edit subnet associations](#)

Find subnet association	
Name	Subnet ID
public-subnet-az-1	subnet-0f0c85ac138a85f07
IPv4 CIDR	10.0.0.0/24
IPv6 CIDR	-

Step 6: create an ec2 instances name it has **Public EC2**

☰ [EC2](#) > [Instances](#) > [Launch an instance](#)

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name
 Add additional tags

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Recentos [Quick Start](#)

       [...>](#)  [Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 kernel-6.1 AMI
ami-00ca570c1b6d79f36 (64-bit (x86), uefi-preferred) / ami-061d45d4bd9c71ba1 (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud application

Amazon Linux 2023 AMI 2023.9.20251208.0 x86_64 HVM kernel-6.1

Architecture	Boot mode	AMI ID	Publish Date	Username (i)
64-bit (x86) ▾	uefi-preferred	ami-00ca570c1b6d79f36	2025-12-03	ec2-user

Verified provider

▼ Instance type [Info](#) | Get advice

Instance type

t2.micro
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0268 USD per Hour
On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0142 USD per Hour
On-Demand SUSE base pricing: 0.0124 USD per Hour

All generations [Compare instance types](#)

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

newly [Create new key pair](#)

▼ Network settings [Info](#)

VPC - required [Info](#)

vpc-0fb771a02f54d53ca (Demo-vpc)
10.0.0.0/16 [Create new VPC](#)

Subnet [Info](#)

subnet-0fc85ac138a85f07 public-subnet-az-1
VPC: vpc-0fb771a02f54d53ca Owner: 801458814717 Availability Zone: ap-south-1a (aps1-az1) Zone type: Availability Zone IP addresses available: 251 CIDR: 10.0.0.0/24 [Create new subnet](#)

Auto-assign public IP [Info](#)

Enable [Create security group](#) [Select existing security group](#)

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

[Create security group](#) [Select existing security group](#)

[Create security group](#) [Select existing security group](#)

Security group name - required

launch-wizard-1

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-./@#=;<>[]\$^

Description - required [Info](#)

launch-wizard-1 created 2025-12-22T12:44:27.846Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type Info	Protocol Info	Port range Info	Remove
ssh	TCP	22	
Source type Info	Source Info	Description - optional Info	
Anywhere	Add CIDR, prefix list or security group	e.g. SSH for admin desktop	
0.0.0.0/0 X			

▼ Security group rule 2 (TCP, 80, 0.0.0.0/0)

Type Info	Protocol Info	Port range Info	Remove
HTTP	TCP	80	

Summary

Number of instances [Info](#)

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.9.2 [read more](#)
ami-00ca570c1b6d79f36

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

[Cancel](#) [Launch instance](#) [Preview code](#)

▼ Configure storage [Info](#)

1x 8 GiB gp3 [Root volume, 3000 IOPS, Not encrypted](#)

Advanced

→ launch Instance

i-0ce1daadce2479651 (Public EC2)

Instance summary [Info](#)

Instance ID i-0ce1daadce2479651	Public IPv4 address 15.206.81.22 open address ↗	Private IPv4 addresses 10.0.0.232
IPv6 address -	Instance state Running	Public DNS -
Hostname type IP name: ip-10-0-0-232.ap-south-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-0-232.ap-south-1.compute.internal	Elastic IP addresses -
Answer private resource DNS name o/p: -	Instance type t2.micro	

Step 7: launch another EC2 Instance for private EC2

→ enable the DNS in vpc

Go to demo-vpc → actions → select edit vpc settings → select enable DNS hostname (save it)

Edit VPC settings [Info](#)

VPC details

VPC ID: [vpc-0fb771a02f54d53ca](#)
Name: [Demo-vpc](#)

DNS settings

DHCP option set: [Info](#)
dopt-0af5547b670e1321b

DNS settings

Enable DNS resolution [Info](#)
 Enable DNS hostnames [Info](#)

Network Address Usage metrics settings

Enable Network Address Usage metrics [Info](#)

[Cancel](#) [Save](#)

Ec2 instance name:

Name : Private EC2

AMI: amazon linux

Instance type: t2.micro

Key pair: select old one

Network settings:

▼ Network settings [Info](#)

VPC - required | [Info](#)

vpc-0fb771a02f54d53ca (Demo-vpc)
10.0.0.0/16

Subnet | [Info](#)

subnet-007d7b77ec9f2370e private-subnet-az-2
VPC: vpc-0fb771a02f54d53ca Owner: 801458814717
Availability Zone: ap-south-1b (aps1-az3) Zone type: Availability Zone
IP addresses available: 251 CIDR: 10.0.1.0/24

Create new subnet [Create new subnet](#)

Auto-assign public IP | [Info](#)

Enable

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Security group name - required

launch-wizard-2

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-:/()#,@[]+=;&;!\$*

Description - required | [Info](#)

launch-wizard-2 created 2025-12-22T12:58:42.565Z

Security group:

type : ssh

Protocol :TCP

Port range:22

Source : custom (10.0.0.0/16) (By setting the security group source to **10.0.0.0/16**, the EC2 instance allows connections only from servers inside the same VPC. Any connection from outside the VPC or from the internet is blocked)

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 10.0.0.0/16)

[Remove](#)

Type | [Info](#)

ssh

Protocol | [Info](#)

TCP

Port range | [Info](#)

22

Source type | [Info](#)

Custom

Source | [Info](#)

Add CIDR, prefix list or security group

Description - optional | [Info](#)

e.g. SSH for admin desktop

10.0.0.0/16 [X](#)

o/p:we got public and private dns names .to access the server we can use DNS names instead of IP addresses

i-024d8a92764e905d3 (Private EC2)

Instance summary

Instance ID i-024d8a92764e905d3	Public IPv4 address 3.111.36.109 open address	Private IPv4 addresses 10.0.1.36
IPv6 address -	Instance state Running	Public DNS ec2-3-111-36-109.ap-south-1.compute.amazonaws.com open address
Hostname type IP name: ip-10-0-1-36.ap-south-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-1-36.ap-south-1.compute.internal	

Step 8: connect public ec2 instance to local

Go to terminal

> cd downloads

>chmod 400 "newly.pem"

>ssh -i "newly.pem" ec2-user@ec2-15-206-81-22.ap-south-1.compute.amazonaws.com

```
durgasanthoshi@Durgas-MacBook-Air downloads % chmod 400 "newly.pem"
[durgasanthoshi@Durgas-MacBook-Air downloads % ssh -i "newly.pem" ec2-user@ec2-15-206-81-22.ap-south-1.compute.amazonaws.com
] The authenticity of host 'ec2-15-206-81-22.ap-south-1.compute.amazonaws.com (15.206.81.22)' can't be established.
ED25519 key fingerprint is SHA256:fFC/qENomEbTaWdx1ksbtNkwjNeVjTgDlYJvIDbOh8Y.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-15-206-81-22.ap-south-1.compute.amazonaws.com' (ED25519) to the list
of known hosts.
'   #_
~\_ #####      Amazon Linux 2023
~~ \####\_
~~  \##|_
~~   \|/  https://aws.amazon.com/linux/amazon-linux-2023
~~   V~' '-->
~~   /_
~~,-. /_
~~,-/ /_
~~,-/ /_
[ec2-user@ip-10-0-0-232 ~]$
```

Step 9: now try to connect private ec2 instance to local

```
durgasanthoshi@Durgas-MacBook-Air ~ % cd downloads
[durgasanthoshi@Durgas-MacBook-Air downloads % chmod 400 "newly.pem"]
[durgasanthoshi@Durgas-MacBook-Air downloads % ssh -i "newly.pem" ec2-user@ec2-3-111-36-109.ap-south-1.compute.amazonaws.com
```

o/p: operation time out error . because private ec2 instance is not having the internet connection

```
ssh: connect to host ec2-3-111-36-109.ap-south-1.compute.amazonaws.com port 22: Operation timed out
durgasanthoshi@Duras-MacBook-Air downloads %
durgasanthoshi@Duras-MacBook-Air downloads %
```

Step 10: now to try to connect public ec2 instance to private ec2 instance

→ we have our key file in local downloads folder. Now we need to copy the key file.prem from your laptop to the public EC2. This allows public EC2 to connect to private EC2

Open a new terminal

```
>cd downloads
```

```
>ls (check the .pem file is there or not)
```

```
>scp -i newly.pem newly.pem ec2-user@15.206.81.22:/home/ec2-user/
```

(scp → copy a file this command copies my key file from my laptop to the public EC2)

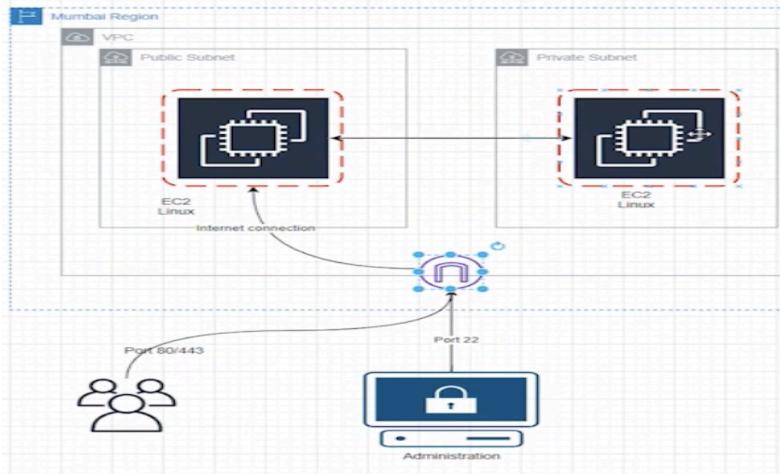
```
durgasanthoshi@Duras-MacBook-Air ~ % cd downloads
durgasanthoshi@Duras-MacBook-Air downloads % ls
ALB.pem
aws load balancer
AWS Roadmap.pdf
aws tasks
Bio data
BIO DATA.pdf
Biodata.pdf
durgasanthoshi@Duras-MacBook-Air downloads % scp -i newly.pem newly.pem ec2-use
r@<PUBLIC-EC2-PUBLIC-IP>:/home/ec2-user/
zsh: no such file or directory: PUBLIC-EC2-PUBLIC-IP
durgasanthoshi@Duras-MacBook-Air downloads % scp -i newly.pem newly.pem ec2-use
r@15.206.81.22:/home/ec2-user/
The authenticity of host '15.206.81.22 (15.206.81.22)' can't be established.
ED25519 key fingerprint is SHA256:fFC/qENomEbTaWdx1ksbtNkwjNeVjTgD1YJvIDbOh8Y.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:40: ec2-15-206-81-22.ap-south-1.compute.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '15.206.81.22' (ED25519) to the list of known hosts.
newly.pem
100% 1674      51.1KB/s   00:00
durgasanthoshi@Duras-MacBook-Air downloads %
```

→ now we have access to connect private ec2 instance in public ec2 instance

```
>copy the private ip address from the ec2 instance
```

```
[ec2-user@ip-10-0-0-232 ~]$ ssh -i "newly.pem" ec2-user@ec2-3-111-36-109.ap-south-1.compute.amazonaws.com
,
#_
~\_ #####_          Amazon Linux 2023
~~ \#####\
~~  \###|
~~   \#/ ___ https://aws.amazon.com/linux/amazon-linux-2023
~~    \~`-->
~~     /
~~.._ _/
~/_ /_/
~/m/_/
[ec2-user@ip-10-0-1-36 ~]$
```

Flow chat :



```
_/m/'  
[ec2-user@ip-10-0-1-36 ~]$ logout  
Connection to ec2-3-111-36-109.ap-south-1.compute.amazonaws.com closed.  
[ec2-user@ip-10-0-0-232 ~]$
```

> logout from the Private ec2 instance in public ec2 instance

Step 11: **run any web server in public ec2 instance**

>install apache server in public ec2 instance

Cmd:

```
>sudo yum install httpd -y
```

```
>sudo systemctl start httpd
```

```
>sudo nano /var/www/html/index.html
```

[Paste the code](#)

```
<!DOCTYPE html>  
<html>  
<head>  
<title>AWS VPC Public EC2</title>
```

```

</head>

<body>

<h1>Public EC2 is Working 🚀</h1>

<p>This web server is running inside a public subnet.</p>

<p>Internet Gateway and Route Table are configured correctly.</p>

</body>

</html>

```



```

GNU nano 8.3                               /var/www/html/index.html
<!DOCTYPE html>
<html>
<head>
    <title>AWS VPC Public EC2</title>
</head>
<body>
    <h1>Public EC2 is Working 🚀</h1>
    <p>This web server is running inside a public subnet.</p>
    <p>Internet Gateway and Route Table are configured correctly.</p>
</body>
</html>

```

→ save the code

→ http://<PUBLIC-EC2-PUBLIC-IP> (paste the public ec2 ipv4 address) (ex:<http://15.206.81.22/>)

o/p:



Public EC2 is Working 🚀

This web server is running inside a public subnet.

Internet Gateway and Route Table are configured correctly.

Finally our web page is running successfully

Summary of what i did:

Reference video :<https://www.youtube.com/watch?v=cyy2ok-00qs>

- ❖ Created **VPC**
- ❖ Created **Public & Private subnets**
- ❖ Attached **Internet Gateway**
- ❖ Configured **Route Tables**
- ❖ Launched **Public EC2 & Private EC2**
- ❖ Connected **Public → Private EC2**
- ❖ Hosted a **web page on Public EC2**

created a VPC with public and private subnets.

The public EC2 is connected to the internet and hosts a web page.

The private EC2 has no internet and can only be accessed securely via the public EC2.

I set up proper route tables, Internet Gateway, and security groups to control access.

Security

Private EC2 is not exposed to the internet

Only accessible from public EC2

Controlled access

Public EC2 acts as a **jump server**

Allows safe internal communication

Can add more private servers without giving them internet

